

# Notes on hierarchical ensemble methods for DAG-structured taxonomies

*Giorgio Valentini*

DI - Dipartimento di Informatica

Università degli Studi di Milano

e-mail: valentini@di.unimi.it

## Abstract

Several real problems ranging from text classification to computational biology are characterized by hierarchical multi-label classification tasks. Most of the methods presented in literature focused on tree-structured taxonomies, but only few on taxonomies structured according to a Directed Acyclic Graph (DAG). In this contribution novel classification ensemble algorithms for DAG-structured taxonomies are introduced. In particular Hierarchical Top-Down (*HTD-DAG*) and True Path Rule (*TPR-DAG*) for DAGs are presented and discussed.

## 1 Introduction

Hierarchical classification problems are characterized by taxonomies structured according to a pre-defined hierarchy. Examples in the context of the gene or protein function prediction include trees or directed acyclic graphs [30], where functional classes are connected according to a tree (FunCat, Functional Categories [27]) or a DAG (GO, Gene Ontology [30]).

Extensive experimental studies showed that flat prediction, i.e. predictions for each class made independently of the other classes, introduce significant inconsistencies in the classification, due to the violation of the *true path rule*, that governs the hierarchical relationships between classes [25, 13]. According to this rule, positive predictions for a given term must be transferred to its “ancestor” terms and negative predictions to its descendants.

In their more general form hierarchical ensemble methods adopt a two-steps learning strategy [23, 14, 10, 28]:

1. In the first step each base learner separately or interacting with connected base learners learns the protein functional category on a per-term basis. In most cases this yields a set of independent classification problems, where each base learning machine is trained to learn a specific functional term, independently of the other base learners.

2. In the second step the predictions provided by the trained classifiers are combined by considering the hierarchical relationships between the base classifiers modeled according to the hierarchy of the functional classes.

Usually they significantly improve prediction performances with respect to “flat” prediction methods [21, 33, 1].

Hierarchical classification and in particular ensemble methods for hierarchical classification have been applied in several domains ranging from protein function prediction [20, 6, 2], to text categorization [24, 15, 36], to music genre classification [7, 18, 31], hierarchical image classification [5, 4] and video annotation [19], and automatic classification of World Wide Web documents [26, 8]. A general review on hierarchical classification methods and their applications in different domains is provided in [29], while a review on hierarchical ensemble methods in the context of computational biology is provided in [34].

Most of the proposed ensemble methods have been proposed for tree-structured taxonomies [9, 11, 33, 22, 14, 12, 13, 16], and only a few for DAG-structured taxonomies [25, 21, 28].

In this contribution we propose and discuss novel hierarchical ensemble methods for DAGs, in particular the Hierarchical Top-Down (*HTD-DAG*), the True Path Rule (*TPR-DAG*) algorithms and some their variants specifically designed for DAG-structured taxonomies.

## 2 Basic notation and definitions

Let  $G = \langle V, E \rangle$  a Directed Acyclic Graph (DAG) with vertices  $V = \{1, 2, \dots, |V|\}$  and edges  $e = (i, j) \in E, i, j \in V$ .  $G$  represents a taxonomy structured as a DAG, whose nodes  $i \in V$  represent classes of the taxonomy and a directed edge  $(i, j) \in E$  the hierarchical relationships between  $i$  and  $j$ :  $i$  is the parent class and  $j$  is the child class. We assume that does exist a unique root node  $root(G)$  of the DAG; if there are multiple roots we can easily add a unique root node by simply adding an edge between the added root and the original multiple roots.

The set of children of a node  $i$  is denoted  $child(i)$ , the set of its parents  $par(i)$ , the set of its ancestors  $anc(i)$  and the set of its descendants  $desc(i)$ .

A “flat continuous” classifier  $f : X \rightarrow \mathbb{Y}$  provides a score  $\hat{\mathbf{y}} \in \mathbb{Y} = [0, 1]^{|V|}$  for a given example  $x \in X$ . In other words a flat classifier provides a score  $\hat{y}_i \in [0, 1]$  for each node/class  $i \in V$  of the DAG  $G$ :

$$\hat{\mathbf{y}} = \langle \hat{y}_1, \hat{y}_2, \dots, \hat{y}_{|V|} \rangle \quad (1)$$

It is easy to see that a “flat discrete” binary classifier is a special case of  $f$ , where  $\hat{y}_i \in \{0, 1\}$ . In this case the classifier simply assigns ( $\hat{y}_i = 1$ ) or not ( $\hat{y}_i = 0$ ) an example  $x$  to class  $i$ . In the more general case the flat “continuous” classifier provides scores  $\hat{y}_i \in [0, 1]$  that can be interpreted as the likelihood or probability of belonging to a given class  $i$ .

In the case of a “flat discrete” classifier the labels  $\hat{\mathbf{y}}$  directly identify the set  $S \subset V$  of the predicted classes:

$$S = \{i | i \in V \wedge \hat{y}_i = 1\} \quad (2)$$

We say that  $S$  is a valid “discrete” labeling if the following property (*true path rule*) holds:

$$S \text{ is valid} \iff S = \{i | i \in V \wedge i \in S \wedge j \in \text{par}(i) \Rightarrow j \in S\} \quad (3)$$

In the more general case of continuous scores, we say that the multi-label scoring  $\mathbf{y}$  is valid if it respects the *true path rule*:

$$\mathbf{y} \text{ is valid} \iff \forall i \in V, i \in \text{par}(j) \Rightarrow y_i \geq y_j \quad (4)$$

It is easy to see that (3) is a special case of (4).

We say that a classifier satisfies the true path rule if for any example  $x$ , for respectively a flat discrete or a continuous classifier property (3) or (4) holds.

In real cases it is very unlikely that a flat discrete or continuous classifier satisfies the true path rule, since by definition the predictions are performed without considering the hierarchy of the classes. Nevertheless by adding a further label/score modification step, i.e. by taking into account the hierarchy of the classes, we can modify the labeling or the scores of the flat classifiers to obtain a hierarchical classifier that obeys the true path rule. In other words we can provide a function  $h(f(x)) : X \rightarrow \mathbb{Y}$  such that  $\forall x \in X$ , given that  $f(x) = \hat{\mathbf{y}}$  and  $h(f(x)) = \bar{\mathbf{y}}$ , we have for all  $(x, \bar{\mathbf{y}})$ :

$$\forall i \in V, i \in \text{par}(j) \Rightarrow \bar{y}_i \geq \bar{y}_j \quad (5)$$

In this contribution we propose several hierarchical algorithms that implement such function  $h$ .

### 3 Hierarchical top-down algorithm for DAGs (HTD-DAG)

The simplest hierarchical algorithm for DAG is the Hierarchical top-down algorithm (*HTD*). It adopts this simple following rule by per-level visiting the nodes from top to bottom:

$$\bar{y}_i = \begin{cases} \hat{y}_i & \text{if } i \in \text{root}(G) \\ \min_{j \in \text{par}(i)} \bar{y}_j & \text{if } \min_{j \in \text{par}(i)} \bar{y}_j < \hat{y}_i \\ \hat{y}_i & \text{otherwise} \end{cases} \quad (6)$$

Note that the level must correspond to the maximum distance of the node from the root.

**Theorem 1** *Given a DAG  $G = \langle V, E \rangle$ , a set of flat predictions  $\hat{\mathbf{y}} = \langle \hat{y}_1, \hat{y}_2, \dots, \hat{y}_{|V|} \rangle$  for each class associated to each node  $i \in \{1, \dots, |V|\}$ , the HTD-DAG algorithm assures that  $\forall i \in V, i \in \text{anc}(j) \Rightarrow \bar{y}_i \geq \hat{y}_i$ .*

The proof by induction is straightforward and omitted for brevity.

Fig. 1 provides the pseudo code of the algorithm. Rows from 1 to 4 provide the distance of each node from the root, where distance means the maximum path length from the node to the root. To this end the classical Bellman-Ford algorithm [17] can be used: by recalling that it finds the shortest paths from a source node to all the other nodes of a weighted digraph, it is sufficient to invert the sign of each edge weight to obtain the maximum distance (longest path) from the root (rows 2 and 3). Note that  $dist$  is a vector containing for each element  $i$  the maximum distance of the node  $i$  from the root.

The second block of the algorithm implements a per-level top-down visit of the graph (rows 5 – 16). Starting from the children of the root (level 1) for each level of the graph the nodes are processed and the hierarchical top-down correction of the flat predictions  $\hat{y}_i$ ,  $i \in \{1, \dots, |V|\}$  to the HTD-DAG ensemble prediction  $\bar{y}_i$  is performed according to eq. 6.

Figure 1: **Hierarchical Top-Down algorithm for DAGs (HTD-DAG)**

```

Input:
-  $G = \langle V, E \rangle$ 
-  $V = \{1, 2, \dots, |V|\}$ 
-  $\hat{\mathbf{y}} = \langle \hat{y}_1, \hat{y}_2, \dots, \hat{y}_{|V|} \rangle$ ,  $\hat{y}_i \in [0, 1]$ 
begin algorithm
01:   A. Compute  $\forall i \in V$  the max distance from  $root(G)$ :
02:      $E' := \{e' | e \in E, e' = -e\}$ 
03:      $G' := \langle V, E' \rangle$ 
04:      $dist := \text{Bellman.Ford}(G', root(G'))$ 
05:   B. Per-level top-down visit of  $G$ :
06:      $\bar{y}_{root(G)} := \hat{y}_{root(G)}$ 
07:     for each  $d$  from 1 to  $\max(dist)$  do
08:        $N_d := \{i | dist(i) = d\}$ 
09:       for each  $i \in N_d$  do
10:          $x := \min_{j \in par(i)} \bar{y}_j$ 
11:         if  $(x < \hat{y}_i)$ 
12:            $\bar{y}_i := x$ 
13:         else
14:            $\bar{y}_i := \hat{y}_i$ 
15:         end for
16:       end for
end algorithm
Output:
-  $\bar{\mathbf{y}} = \langle \bar{y}_1, \bar{y}_2, \dots, \bar{y}_{|V|} \rangle$ 

```

The first block  $A$  (rows 1 – 4) of the algorithm is dominated by the Bellman-Ford algorithm and has complexity  $\mathcal{O}(|V|^2)$  for sparse graphs, while it is easy to see that the complexity of the second block  $B$  (rows 5 – 16) is linear in the number of vertices for sparse graphs.

## 4 Hierarchical True Path Rule algorithm for DAGs (TPR-DAG)

This second algorithm represents a DAG extension of the *TPR* algorithm, originally proposed for tree-structured taxonomies [32, 33]. The main difference with respect to the original tree-version consists in the fact that the per-level traversal is now performed through two completely distinct steps: a bottom-up per level visit of the graph followed by a top-down visit, while in the original tree-version the per-level traversal is performed in an “interleaved” fashion (that is the bottom-up and top-down traversal are alternated at each level [33]). In the DAG version the separation of the bottom-up and top-down steps is necessary to assure the true path rule consistency of the predictions.

The other main difference consists in the way the levels are computed: in this new DAG version the levels are constructed according to the maximum distance from the root, since this guarantees that in the top-down step all the ancestor nodes have been just processed, thus assuring the true path rule consistency of the predictions.

Similarly to the tree-based version the *TPR* algorithm, the *TPR-DAG* adopts a per-level bottom-up traversal of the DAG, starting from the nodes most distant (in the sense of the maximum distance) from the root to correct the flat predictions  $\hat{y}_i$ :

$$\bar{y}_i := \frac{1}{1 + |\phi_i|} (\hat{y}_i + \sum_{j \in \phi_i} \bar{y}_j) \quad (7)$$

where  $\phi_i$  are the “positive” children of  $i$ .

$$\phi_i := \{j \in \text{child}(i) | \bar{y}_j > t_j\} \quad (8)$$

The choice of the set of the “positive” children  $\phi_i$ , that is of the children responsible for the bottom-up propagation of the positive predictions, depends critically on the choice of the threshold  $t_j$ . Possible choices could be the following:

- (a)  $t_j = \bar{t}$ ,  $\forall j \in V$ : that is an equal threshold  $\bar{t}$  is selected for all the nodes. For instance if the predictions represent probabilities it could be meaningful to a priori select  $\bar{t} = 0.5$ .
- (b)  $t_j$  is selected to maximize the F-score estimated on the training data
- (c)  $t_j$  is selected on the basis of the distribution of the positive examples available for the training set (e.g.  $t_j$  could correspond to the  $k^{\text{th}}$  percentile of the positive examples distribution).

Figure 2: **Hierarchical True Path Rule algorithm for DAGs (TPR-DAG)**

Input:

- $G = \langle V, E \rangle$
- $V = \{1, 2, \dots, |V|\}$
- $\hat{\mathbf{y}} = \langle \hat{y}_1, \hat{y}_2, \dots, \hat{y}_{|V|} \rangle$ ,  $\hat{y}_i \in [0, 1]$
- $\mathbf{t} = \langle t_1, t_2, \dots, t_{|V|} \rangle$ ,  $t_i \in [0, 1]$

begin algorithm

01: A. Compute  $\forall i \in V$  the max distance from  $root(G)$ :

02:  $E' := \{e' | e \in E, e' = -e\}$

03:  $G' := \langle V, E' \rangle$

04:  $dist := \text{Bellman.Ford}(G', root(G'))$

05: B. Per-level bottom-up visit of  $G$ :

06: for each  $d$  from  $\max(dist)$  to 1 do

07:  $N_d := \{i | dist(i) = d\}$

08: for each  $i \in N_d$  do

09:  $\phi_i := \{j \in child(i) | \bar{y}_j > t_j\}$

10:  $\bar{y}_i := \frac{1}{1+|\phi_i|}(\hat{y}_i + \sum_{j \in \phi_i} \bar{y}_j)$

11: end for

12: end for

13: C. Per-level top-down visit of  $G$ :

14:  $\bar{y}_{root(G)} := \hat{y}_{root(G)}$

15: for each  $d$  from 1 to  $\max(dist)$  do

16:  $N_d := \{i | dist(i) = d\}$

17: for each  $i \in N_d$  do

18:  $x := \min_{j \in par(i)} \bar{y}_j$

19: if  $(x < \hat{y}_i)$

20:  $\bar{y}_i := x$

21: else

22:  $\bar{y}_i := \hat{y}_i$

23: end for

24: end for

end algorithm

Output:

- $\bar{\mathbf{y}} = \langle \bar{y}_1, \bar{y}_2, \dots, \bar{y}_{|V|} \rangle$

A different solution, that does not require an a priori or and experimentally selected threshold could consists in choosing those children that can increment the score of the node  $i$  (that is positive nodes are those that achieve a higher score than that of their parent):

$$\phi_i := \{j \in child(i) | \bar{y}_j > \hat{y}_i\} \quad (9)$$

Independently of the choice of the positive children, the following theorem holds:

**Theorem 2** *Given a DAG  $G = \langle V, E \rangle$ , a set of flat predictions  $\hat{\mathbf{y}} = \langle \hat{y}_1, \hat{y}_2, \dots, \hat{y}_{|V|} \rangle$  for each class associated to each node  $i \in \{1, \dots, |V|\}$ , the TPR-DAG algorithm assures that  $\forall i \in V, i \in \text{anc}(j) \Rightarrow \bar{y}_i \geq \bar{y}_j$ .*

The proof is omitted for brevity.

Fig. 2 shows the high-level pseudo-code of the *TPR-DAG* algorithm. The first four rows compute the maximum distance of each node from the root, using the Bellman-Ford algorithm. The block *B* (rows 5-12) performs a bottom-up visit of the graph and updates the predictions  $\bar{y}_i$  of the *TPR-DAG* ensemble according to eq. 7 and 8. Note that this step propagates the “positive” predictions from bottom to top of the DAG, but does not assure the true path rule consistency of the predictions. This is accomplished by the third block (rows 13 – 24) that simply executes a hierarchical top-down step, in the same way of the *HTD-DAG* algorithm.

Also for the *TPR-DAG* algorithm the complexity is quadratic in the number of nodes for the block *A*, and it is easy to see that it is  $\mathcal{O}(|V|)$  for both the *B* and *C* blocks when graphs are sparse.

A variant of the *TPR* algorithm for DAG is represented by the *ISO-TPR* algorithm (Fig. 3). The main difference with respect to the *TPR-DAG* algorithm is constituted by the top-down step: instead of using the *HTD-DAG* top-down step a partial order isotonic regression approach is applied [3]. That is, the following optimization problem is solved (rows 13-14):

$$\bar{\mathbf{y}} = \begin{cases} \min_{\bar{\mathbf{y}}} \sum_{i \in V} (\hat{y}_i - \bar{y}_i)^2 \\ \forall i, \quad j \in \text{par}(i) \Rightarrow \bar{y}_i \geq \bar{y}_j \end{cases} \quad (10)$$

In this way the true path rule constraints are maintained by construction, and the solution closest to flat predictions (in the sense of the squared error) that obeys the true path rule is selected.

It is worth noting that other variants similar to the weighted True Path Rule algorithms for tree-structured taxonomies [35, 13] can be designed for DAGs, simply by substituting row 10 of the *TPR-DAG* algorithm (Fig. 2). For instance we can obtain the *TPR-w-DAG* algorithm by substituting row 10 of the *TPR-DAG* algorithm with the following line of pseudocode:

$$\bar{y}_i := w\hat{y}_i + \frac{(1-w)}{|\phi_i|} \sum_{j \in \phi_i} \bar{y}_j \quad (11)$$

In this approach a weight  $w \in [0, 1]$  is added to balance between the contribution of the node  $i$  and that of its “positive” children.

As shown in [33], the contribution of the descendants of a given node decays exponentially with their distance from the node itself. To enhance the contribution of the most specific nodes to the overall decision of the ensemble a linear decaying or a constant contribution of the “positive” descendants could be considered instead:

$$\bar{y}_i := \frac{1}{1 + |\Delta_i|} (\hat{y}_i + \sum_{j \in \Delta_i} \bar{y}_j) \quad (12)$$

Figure 3: **Hierarchical Isotonic True Path Rule algorithm for DAGs (ISO-TPR)**

Input:

- $G = \langle V, E \rangle$
- $V = \{1, 2, \dots, |V|\}$
- $\hat{\mathbf{y}} = \langle \hat{y}_1, \hat{y}_2, \dots, \hat{y}_{|V|} \rangle, \quad \hat{y}_i \in [0, 1]$
- $\mathbf{t} = \langle t_1, t_2, \dots, t_{|V|} \rangle, \quad t_i \in [0, 1]$

begin algorithm

- 01:   A. Compute  $\forall i \in V$  the max distance from  $root(G)$ :
- 02:      $E' := \{e' | e \in E, e' = -e\}$
- 03:      $G' := \langle V, E' \rangle$
- 04:      $dist := \text{Bellman.Ford}(G', root(G'))$
- 05:   B. Per-level bottom-up visit of  $G$ :
- 06:     for each  $d$  from  $\max(dist)$  to 1 do
- 07:        $N_d := \{i | dist(i) = d\}$
- 08:       for each  $i \in N_d$  do
- 09:           $\phi_i := \{j \in child(i) | \bar{y}_j > t_j\}$
- 10:           $\bar{y}_i := \frac{1}{1+|\phi_i|}(\hat{y}_i + \sum_{j \in \phi_i} \bar{y}_j)$
- 11:       end for
- 12:     end for
- 13:   C. Isotonic correction:
- 14:      $\bar{\mathbf{y}} = \begin{cases} \min_{\bar{\mathbf{y}}} \sum_{i \in V} (\hat{y}_i - \bar{y}_i)^2 \\ \forall i, \quad j \in par(i) \Rightarrow \bar{y}_j \geq \bar{y}_i \end{cases}$

end algorithm

Output:

- $\bar{\mathbf{y}} = \langle \bar{y}_1, \bar{y}_2, \dots, \bar{y}_{|V|} \rangle$

where

$$\Delta_i = \{j \in desc(i) | \bar{y}_j > t_j\} \quad (13)$$

In this way all the “positive” descendants of node  $i$  provide the same contribution to the ensemble prediction  $\bar{y}_i$ . Analogously, we can design “positive” descendants whose contributions to  $\bar{y}_i$  decays linearly with their distance from the root.

## 5 Conclusions

The *HTD-DAG* algorithm represents a simple and efficient approach to make consistent the predictions of flat classifiers in a DAG-structured taxonomy. The *TPR-DAG* and its variants including *ISO-TPR* represent an adaptation to DAGs of the previously proposed *TPR* algorithm for tree-structured taxonomies. *TPR-DAG* exploits the children (and



recursively the descendants) of each node of the DAG-structured ensemble, to propagate from bottom to top the positive predictions of the classifiers associated to the most specific classes of the hierarchy. In this way positive predictions for the most specific classes are valued and passed to the less specific ones. A second top-down step re-establishes the true path rule constraints and propagate negative predictions towards the descendants.

Experimental studies with DAG-structured taxonomies are planned to assess the effectiveness of the proposed methods for real-world hierarchical classification problems.

## References

- [1] N. Alaydie, C.K. Reddy, and F. Fotouhi. Exploiting Label Dependency for Hierarchical Multi-label Classification. In *Advances in Knowledge Discovery and Data Mining*, number 7301 in Lecture Notes in Computer Science, pages 294–305. Springer-Verlag, 2012.
- [2] K. Astikainen, L. Holm, E. Pitkanen, S. Szedmak, and J. Rousu. Towards structured output prediction of enzyme function. *BMC Proceedings*, 2(Suppl 4:S2), 2008.
- [3] R.E. Barlow and H.D. Brunk. The isotonic regression problem and its dual. *Journal of the American Statistical Association*, 67(337):140–147, 1972.
- [4] Z. Barutcuoglu and C. DeCoro. Hierarchical shape classification using bayesian aggregation. In *Proc. of the IEEE Conf. on Shape Modeling and Applications*, 2006.
- [5] A. Binder, M. Kawanabe, and U. Brefeld. Bayesian aggregation for hierarchical genre classification. In *Proc. of the 9th Asian Conf. on Computer Vision*, 2009.
- [6] H. Blockeel, L. Schietgat, and A. Clare. Hierarchical multilabel classification trees for gene function prediction. In J. Rousu, S. Kaski, and E. Ukkonen, editors, *Probabilistic Modeling and Machine Learning in Structural and Systems Biology*, Tuusula, Finland, 2006. Helsinki University Printing House.
- [7] A. Burred, J. and Lerch. A hierarchical approach to automatic musical genre classification. In *Proc. of the 6th Int. Conf. on Digital Audio Effects*, pages 8–11, 2003.
- [8] M. Ceci and D. Malerba. Classifying web documents in a hierarchy of categories: A comprehensive study. *Journal of Intelligent Information Systems*, 28(1):1–41, 2007.
- [9] R. Cerri and A. de Carvalho. Hierarchical multilabel classification using top-down label combination and artificial neural networks. In *SBRN '10 Proceedings of the 2010 Eleventh Brazilian Symposium on Neural Networks*, pages 253–258. IEEE Computer Society, 2010.
- [10] R. Cerri and A. de Carvalho. New top-down methods using SVMs for hierarchical multilabel classification problems. In *IJCNN 2010*, pages 1–8. IEEE Computer Society, 2010.

- [11] R. Cerri and A. de Carvalho. Hierarchical multilabel protein function prediction using local neural networks. In *Advances in Bioinformatics and Computational Biology*, number 6832 in Lecture Notes in Computer Science, pages 10–17. Springer-Verlag, 2011.
- [12] N. Cesa-Bianchi, C. Gentile, and L. Zaniboni. Hierarchical classification: Combining Bayes with SVM. In *Proc. of the 23rd Int. Conf. on Machine Learning*, pages 177–184. ACM Press, 2006.
- [13] N. Cesa-Bianchi, M. Re, and G. Valentini. Synergy of multi-label hierarchical ensembles, data fusion, and cost-sensitive methods for gene functional inference. *Machine Learning*, 88(1):209–241, 2012.
- [14] N. Cesa-Bianchi and G. Valentini. Hierarchical cost-sensitive algorithms for genome-wide gene function prediction. *Journal of Machine Learning Research, W&C Proceedings, Machine Learning in Systems Biology*, 8:14–29, 2010.
- [15] S. Chakrabarti, B. Dom, R. Agrawal, and P. Raghavan. Scalable feature selection, classification and signature generation for organizing large text databases into hierarchical topic taxonomies. *The VLDB Journal*, 7:163–178, 1998.
- [16] B. Chen and J. Hu. Hierarchical multi-label classification based on over-sampling and hierarchy constraint for gene function prediction. *IEEE Transactions on Electrical and Electronic Engineering*, 7:183–189, 2012.
- [17] TH Cormen, CE Leiserson, and RL Rivest. *Introduction to Algorithms*. MIT Press, Boston, 2009.
- [18] C. DeCoro, Z. Barutcuoglu, and R. Fiebrink. Bayesian aggregation for hierarchical genre classification. In *Proc. of the 8th Int. Conf. on Music Information Retrieval*, pages 77–80, 2007.
- [19] A. Dimou, G. Tsoumakas, V. Mezaris, I. Kompatsiaris, and I. Vlahavas. An empirical study of multi-label methods for video annotation. In *Proc. 7th International Workshop on Content-Based Multimedia Indexing, CBMI 09*, Chania, Greece, 2009.
- [20] R. Eisner, B. Poulin, D. Szafron, and P. Lu. Improving protein prediction using the hierarchical structure of the Gene Ontology. In *IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, 2005.
- [21] Y Guan, C.L. Myers, D.C. Hess, Z. Barutcuoglu, A. Caudy, and O.G. Troyanskaya. Predicting gene function in a hierarchical context with an ensemble of classifiers. *Genome Biology*, 9(S2), 2008.
- [22] J. Hernandez, L.E. Sucar, and E. Morales. A hybrid global-local approach for hierarchical classification. In *Proc. of the Twenty-Sixth International Florida Artificial Intelligence Research Society Conference*, pages 432–437, 2013.

- [23] X. Jiang, N. Nariai, M. Steffen, S. Kasif, and E. Kolaczyk. Integration of relational and hierarchical network information for protein function prediction. *BMC Bioinformatics*, 9(350), 2008.
- [24] D. Koller and M. Sahami. Hierarchically classifying documents using very few words. In *Proc. of the 14th Int. Conf. on Machine Learning*, pages 170–178, 1997.
- [25] G. Obozinski, G. Lanckriet, C. Grant, Jordan. M., and W.S. Noble. Consistent probabilistic output for protein function prediction. *Genome Biology*, 9(S6), 2008.
- [26] K. Punera and J. Ghosh. Enhanced hierarchical classification via isotonic smoothing. In *WWW 2008*, pages 151–160, Beijing, China, 2008. ACM.
- [27] A. Ruepp, A. Zollner, D. Maier, K. Albermann, J. Hani, M. Mokrejs, I. Tetko, U. Guldener, G. Mannhaupt, M. Munsterkotter, and H.W. Mewes. The FunCat, a functional annotation scheme for systematic classification of proteins from whole genomes. *Nucleic Acids Research*, 32(18):5539–5545, 2004.
- [28] L. Schietgat, C. Vens, J. Struyf, H. Blockeel, and S. Dzeroski. Predicting gene function using hierarchical multi-label decision tree ensembles. *BMC Bioinformatics*, 11(2), 2010.
- [29] C. Silla and A. Freitas. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1-2):31–72, 2011.
- [30] The Gene Ontology Consortium. Gene ontology: tool for the unification of biology. *Nature Genet.*, 25:25–29, 2000.
- [31] K. Trohidis, G. Tsoumahas, G. Kalliris, and I.P. Vlahavas. Multilabel classification of music into emotions. In *Proc. of the 9th International Conference on Music Information Retrieval*, pages 325–330, 2008.
- [32] G. Valentini. True path rule hierarchical ensembles. In J. Kittler, J. Benediktsson, and F. Roli, editors, *Multiple Classifier Systems. Eighth International Workshop, MCS 2009, Reykjavik, Iceland*, volume 5519 of *Lecture Notes in Computer Science*, pages 232–241. Springer, 2009.
- [33] G. Valentini. True Path Rule hierarchical ensembles for genome-wide gene function prediction. *IEEE ACM Transactions on Computational Biology and Bioinformatics*, 8(3):832–847, 2011.
- [34] G. Valentini. Hierarchical Ensemble Methods for Protein Function Prediction. *ISRN Bioinformatics*, 2014(Article ID 901419):34 pages, 2014.
- [35] G. Valentini and M. Re. Weighted True Path Rule: a multilabel hierarchical algorithm for gene function prediction. In *MLD-ECML 2009, 1st International Workshop on learning from Multi-Label Data*, pages 133–146, Bled, Slovenia, 2009.

- [36] M.L. Zhang and Z.H. Zhou. Multi-label neural network with applications to functional genomics and text categorization. *IEEE Trans. on Knowledge and Data Engineering*, 18(10):1338–1351, 2006.